**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**імені ІГОРЯ СІКОРСЬКОГО»**

**Інститут телекомунікаційних систем**

**Кафедра Інформаційно-телекомунікаційних мереж**

До захисту допущено:
Завідувач кафедри
_____ Лариса ГЛОБА
«___»_____2020 р.

# Дипломна робота

**на здобуття ступеня бакалавра**
**за освітньо-професійною програмою «Інформаційно-комунікаційні технології»**
**спеціальності 172 «Телекомунікації та радіотехніка»**
**на тему: «Метод захисту інформації в мережі інтернету речей»**

Виконав :
студент IV курсу, групи ТІ-61
Безвугляк Максим Сергійович _____

Керівник:
асистент кафедри ІТМ ІТС,
Курдеча В.В. _____

Рецензент:
Посада, науковий ступінь, вчене звання,
Прізвище, ім’я, по батькові _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____

Київ – 2020 року

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Інститут телекомунікаційних систем**

**Кафедра Інформаційно-телекомунікаційних мереж**

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 172 «Телекомунікації та радіотехніка»

Освітньо-професійна програма «Інформаційно-комунікаційні технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Лариса ГЛОБА

«___»_____2020 р.

# ЗАВДАННЯ

**на дипломну роботу студенту**

**Безвугляку Максиму Сергійовичу**

1. Тема роботи «Метод захисту інформації в мережі інтернету речей», керівник роботи ассистент кафедри інформаційно-телекомунікаційних мереж Курдеча Василь Васильович, затверджені наказом по університету від «30» березня 2020 р. № 924-с

2. Термін подання студентом роботи 8 червня 2020 р.

3. Вихідні дані до роботи: використання REST API в мережі інтернету речей, наукові статті про захист інформації в мережі, та криптографії.

4. Зміст роботи (перелік завдань, які потрібно розробити):

1. Провести аналіз проблем захисту інформації в мережах інтернету речей
2. Проаналізувати існуючі способи захисту інформації та вибрати прототип
3. Удосконалити метод захисту інформації за рахунок впровадження REST API як проміжного програмного забезпечення та еліптичної криптографії.

4. Провести натурне моделювання та аналітичну оцінку запропонованого рішення

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо):

- Титульний слайд
- Актуальність
- Мета, основні задачу
- Проблеми захисту інформації в мережі інтернету речей
- Вибір програмного забезпечення як прототипу
- Таблиця оцінки порівнянняпрограмного забезпечення
- Огляд REST API
- Запропонований метод
- Натурне моделювання
- Методи для порівняння
- Результати натурного моделювання
- Результати аналітичної оцінки
- Загальні висновки

6. Дата видачі завдання 9 жовтня 2019 року

Календарний план

| № з/п | Назва етапів виконання дипломної роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Огляд проблеми захисту інформації в мережі інтернету речей. | 16.10.2019-20.11.2019 | виконано |
| 2 | Вибір прототипу на основі існуючих рішень захисту інформації | 21.11.2019-28.12.2019 | виконано |
| 3 | Огляд та аналіз використання REST API в мережі інтернету речей | 6.01.2020 – 20.01.2020 | виконано |
| 4 | Написання наукової статті | 05.02.2019-13.04.2020 | виконано |

| | | | |
|---|---|---|---|
| | (ПТ-2020). | | |
| 5 | Створення запропонованого методу захисту інформації в мережі інтернету речей . | 10.02.2020 – 15.03.2020 | виконано |
| 6 | Проведення натурного моделювання запропонованого методу. | 17.04.2019-30.04.2020 | виконано |
| 7 | Проведення аналізу запропонованого методу | 01.05.2019-11.05.2020 | виконано |
| 8 | Підготовка тексту диплома | 12.05.2020 – 08.06.2020 | виконано |

Студент Максим БЕЗВУГЛЯК

Керівник Василь КУРДЕЧА

# РЕФЕРАТ

Робота містить 64 сторінки, 13 рисунків, 4 таблиці. Було використано 17 джерел.

**Актуальність:** актуальність дослідження полягає у тому, що з кожним роком кількість підключених пристроїв до мережі інтернету речей, які збирають інформацію зростає. Постає необхідність захищати передачу даних, тому що технології зловмисників з кожним роком розвиваються та їм стає все легче перехоплювати передану інформації.

Зараз у світі постає тенденція швидкості передачі інформації, паралельно цьому все гостріше постає питання захисту інформації. Тому в роботі буде запропоновано метод, який підвищує рівень захисту інформації не погіршуючи значно показник швидкості.

**Мета роботи:** підвищити рівень захисту інформації в мережі інтернету речей. В ході виконання цієї роботи проведено аналіз особливостей мережі інтернету речей. Запропонований метод дозволить підвищити рівень захисту в мережі інтернету речей.

Створений метод викоритсовує REST API як проміжне програмне забезпечення та алгоритми шифрування AES256, та алгоритм Еліптичної криптографії.

**Ключові слова:** інтернет речей, захист інформації, еліптична криптографія, REST API

## ABSTRACT

The work contains 64 pages, 13 figures, 4 tables. 17 sources were used.

**Topicality:** the relevance of the study is that every year the number of devices connected to the Internet of Things that collect information is growing. There is a need to protect data transmission, because the technologies of attackers are evolving every year and it is becoming easier for them to intercept the transmitted information.

Now in the world there is a trend of speed of information transfer, at the same time the question of information protection becomes more and more acute. Therefore, the paper will propose a method that increases the level of information protection without significantly worsening the speed.

**The goal of the work:** increase the level of protection of information on the Internet of Things. In the course of this work, an analysis of the features of the Internet of Things was carried out. The proposed method will increase the level of protection in the Internet of Things.

The created method uses the REST API as intermediate software and AES256 encryption algorithms, and the Elliptical Cryptography algorithm.


**Keywords:** Internet of Things, Information protection, Elliptical cryptography, REST API

# Зміст

**ВСТУП** ..... 9

**РОЗДІЛ 1.** ..... 11

**АНАЛІЗ ПРОБЛЕМ ЗАХИСТУ ІНФОРМАЦІЇ В МЕРЕЖАХ ІОТ ТА ІСНУЮЧИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ.** ..... 11

1.1 Огляд проблеми захисту інформації в мережі інтернету речей ..... 11

1.2 Основні принципи та характеристики мережі інтернету речей. ..... 15

1.3 Вибір прототипу на основі існуючих рішень ..... 27

1.4 Огляд застосування REST API за основу прототипу. ..... 34

Висновки ..... 37

**РОЗДІЛ 2.** ..... 38

**УДОСКОНАЛЕНИЙ МЕТОД ЗАХИСТУ ІНФОРМАЦІЇ ЗА РАХУНОК ВПРОВАДЖЕННЯ REST API ЯК ПРОМІЖНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ЕЛІПТИЧНОЇ КРИПТОГРАФІЇ** ..... 38

2.1 Метод застосування REST API в мережі IoT ..... 38

2.2 Модифікований метод захисту інформації ..... 41

Висновки ..... 48

**РОЗДІЛ 3.** ..... 49

**ПРОВЕДЕННЯ НАТУРНОГО МОДЕЛЮВАННЯ МЕТОДУ ТА НАДАННЯ АНАЛІТИЧНОЇ ОЦІНКИ ЗАПРОПОНОВАНОГО МЕТОДУ** ..... 49

3.1 Натурне моделювання ..... 49

3.2 Аналітична оцінка запропонованого рішення ..... 59

Висновки ..... 60

**ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ** ..... 61

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ** ..... 62

# ПЕРЕЛІК СКОРОЧЕНЬ

| | |
|---|---|
| IoT | Internet of Things |
| DDOS | Destributed Denial of System |
| REST | Representational State Transfer |
| SOAP | Simple Object Access Protocol |
| JSON | JavaScript Object Notation |
| XML | eXtensible Markup Language |
| API | Application Programming Interface |
| RPC | Remote Procedure Call |
| GSM | Groupe Spécial Mobile |
| UMTC | Universal Mobile Telecommunications System |
| HTTPS | HyperText Transfer Protocol Secure |
| SSL | Secure Sockets Layer |
| TLS | Transport Layer Security |

# ВСТУП

**Актуальність.** актуальність дослідження полягає у тому, що з кожним роком збільшується кількість пристроїв підключених до мережі IoT, які передають дані, що потребують захисту.

Ваші підключені пристрої є збирачами та передавачами великої кількості даних. Особиста інформація, зібрана та зберігається за допомогою цих пристроїв - наприклад, ваше ім'я, вік, дані про стан здоров'я, місцезнаходження та багато іншого - може допомогти злочинцям у крадіжці вашої особи.

**Мета й завдання дослідження:**

**Метою роботи** є підвищення захисту інформації в мережі інтернету речей за рахунок удосконалення методу захисту інформації на основі REST API.

**Для досягнення мети дослідження було поставлено та вирішено такі основні задачі:**

1. Провести аналіз проблем захисту інформації в мережах інтернету речей
2. Проаналізувати існуючі способи захисту інформації та вибрати прототип
3. Удосконалити метод захисту інформації за рахунок впровадження REST API як проміжного програмного забезпечення та еліптичної криптографії.
4. Провести натурне моделювання та аналітичну оцінку запропонованого рішення

**Об'єкт роботи**: процес захисту інформації в мережі Інтернету речей

**Предмет дослідження**: метод захисту інформації на основі REST API

**Теоретичний результат дослідження**: Запропонований метод захисту інформації, оснований на REST API як проміжне програмне забезпечення, та алгоритму еліптичної криптографії.

**Практичний результат роботи**: Розроблене програмне забезпечення, яке може використовуватися в комплексі технологій мережі інтернету речей, для підвищення рівню захисту інформації.

**Публікації**

1. Безвугляк М.С. Захист інформації в мережі інтернету речей на основі REST API // XIV Міжнародна Науково-технічна Конференція "Проблеми телекомунікацій 2020"
2. Безвугляк М.С., Курдеча В.В Захист інформації в мережі інтернету речей на основі REST API // XIV Міжнародна Науково-технічна Конференція студентів та аспірантів "Перспективи розвитку інформаційно-телекомунікаційних технологій та систем-2020"

# РОЗДІЛ 1.

# АНАЛІЗ ПРОБЛЕМ ЗАХИСТУ ІНФОРМАЦІЇ В МЕРЕЖАХ ІОТ ТА ІСНУЮЧИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ.

## 1.1 Огляд проблеми захисту інформації в мережі інтернету речей

Прогнозується, що до 2021 р. кількість пристроїв інтегрованих в середі Інтернету речей перебільшить один мільярд. Поруч з інтенсивним розповсюдженням по всьому світі та широким застосуванням цієї технології в багатьох областях виробництва та життєзабезпечення гостро стає питання про безпеку не тільки рядових користувачів цієї технології, але критично важливої інформації, циркулюючої при взаємодії між машинами

Через меншу ємність пам’яті, пам’яті та можливість обробки, багатьох пристроїв IoT можуть діяти на меншій потужності, а значить, і на безпеку невдалих заходів, і пристрої стають тими жертвами криптографічних процесів, які безпечно можуть передавати інформацію з відомою тривалістю. Ці засоби дуже схильні атаки аналізу потужності які можуть бути використані для зворотного зміщення цієї формули. На примусові доцільні засоби потрібно використовувати тільки швидко, для обґрунтування процесів кодування.

Кіберзлочинці можуть легко атакувати пристрої IoT завдяки доступу до конфігурації програмного забезпечення за замовчуванням, нерегулярні оновлення встановленого програмного забезпечення, маючи великий розрив між ними. Кіберзлочинці можуть мати доступ до кожного пристрою через типовий режим уразливості даних для входу. BOT NET, mirai зламав багато пристроїв таким чином. Ризик можна зменшити, змінивши заводські ім'я та пароль. Ще одна проблема безпеки BOTNET - це викупна інфекція [6].

Він замикає пристрій за допомогою шифрування і доступ до нього можна отримати лише після оплати викупу. Вторгнення та виявлення системи також відіграють неабияку роль в захисті пристроїв IoT від атак DDOS. Більшість IoT пристроїв підключаються через telnet, який є основним аспектом злочину. Незахищений Інтернет-протокол із використанням інструментів для сканування в Інтернеті, такі як Zmap, Nmap. інформація може бути легко отриманою. Вторгнення Система виявлення може використовуватися для зменшення кібератак.

Щоб збалансувати ці обмеження, різні шари оборони, відокремлювалюють пристрої на окремі мережі, що використовують брандмауери. Перевірка і санкціонування пристрїв IoT досить не простий процес, тому що пристрій повинен довести своє визнання, перш ніж вони можуть відкрити джерела. Іноді, різні пристрої IOT провалюються в цьому процесі.

У ряді труднощів можна побачити оновлення, патчі безпеки застосовуються до коду або програмного забезпечення які функціонують на пристроях IoT. Для Наприклад, щоб слідкувати за доступними оновленнями, ми потрібно застосовувати оновлення без перерв на різні середовища та гетерогенні пристрої, які підключитися через ряд мережних протоколів

До підтримки оновлення без простоїв необхідно, щоб пристрої були в робочому стані або на наступну тривалість часу для створення форми.

Веб, мобільні та хмарні програми і послуги, що використовуються для переговорів, підходу та обчислення параметрів IoT та інформацію, так що їх слід захищати як члени різного рівня системи безпеки IoT. Порушення підключення, поломка пристрою або збільшення кількості службових атак викликають порушення системи.

Одною з складних задач в розвитку концепції інтернету речей в багатьох додатках являється складні проблеми забезпечення інформаційної безпеки в

широкому спектрі захисту від загроз зловмисника. Ці проблеми являються особливо актуальними, оскільки прогнозується ріст потреби користувачів в Інтернеті речей. Для аналізу буде взята одна з найбільш розповсюджених архітектур Інтернету речей, яка складається з трьох рівнів – рівень сприйняття, мережевий та прикладний рівень.

Для кожного з цих рівнів приведені основні проблеми забезпечення інформаційної безпеки. Відмічаються основні причини складності забезпечення інформаційної безпеки на мережевому рівні - гетерогенний характер структури та велика кількість об'єктів Інтернету речей. Система інтернету речей приймає інформацію від великої кількості та збирає великі данні різних форматів від багатьох джерел з неоднорідними характеристиками. Результатом цього являються відмови DoS з-за перевантажень мережі, програмні помилки з-за складності відладки в реальному масштабі часу за допомогою імітатора зовнішнього навантаження.

Проблеми інформаційної безпеки на рівнях структури інтернету речей. Треба відмітити, що в деяких, що в деяких роботах розглядається більш детальна система, в нашій ми будемо розглядати лише три рівні.

Проблеми інформаційної безпеки на рівні сприйняття. Основною проблемою безпеки на рівні сприйняття залежить в фізичній безпеці приборів сприйняття и безпеці збору інформації. Більшість вузлів сприйняття, для яких характерне розгортання в необсуговуючій людиной середі при відсутності стандартів, простота, обмеження енергозабезпечення та слабка здатність до захисту безпеки.

Тому IoT не може забезпечити уніфіковану систему захисту безпеки та являється вразливою до загроз зловмисника. Так як безпровідна сенсорна мережа на рівні сприйняття являється джерелом інформації, то інформаційна безпека на цьому рівні дуже важлива.

Проблеми безпеки на цьому рівні включає фізичний захват сенсорних вузлів, захват шлюзу, витік інформації сенсору, загрози цілісності даних, виснаження енергозабезпечення, загрози перевантаження, атаки типу DoS, загрози маршрутизації встановленням в мережу нелегітимних сенсорів, та загрози копіювання вузла.

Проблеми інформаційної безпеки на мережевому рівні. Загрози інформаційної безпеки існуючих мережей зв'язку розповсюджуються на IoT, який побудований на них. Це відноситься до несанкціонованого доступу, перехвату даних, конфеденційності, DoS атак, вірусами, експлойтами, мережевими. Крім того, існують міжмережеві проблеми аутентифікації, які можуть бути причинами атак Dos.

В IoT стоять більш складні проблеми забезпечення безпеки. Окрема увага надається вразливостям програмного забезпечення, які приводять до порушення інформаційної безпеки після впровадження. Причинами програмної вразливості можуть бути помилки розробників складного багатошарового програмного забезпечення, помилки ядра програми, неповнота обробки виключень, застосування не захищеного коду, не оброблених масивів с здатністю їх перевантаження зловмисниками, помилки обробки великих даних, помилки БД. Важливо відмітити складність ПО в IoT, викликану великим різновидом використаних апаратних платформ та операційних систем[4].

Для проектування ПЗ потрібно емулювати повндінку приборів IoT , створити імітатор зовнішньої середи для серверів. По причині обмеження в приладах в мережі інтернету речей стоїть складна задача обійти велику відмінність між емулятором та прибором. Також для отгрузки відлаженого робочого релізу додатку, необхідно провести повноцінне тестування, включаючи тестування навантаження, тестування продуктивності та комплексне тестування взаємодії модулів.

Іншою причиною програмної вразливості можуть бути БЄКДОРИ це участки кода, внесені розробником, для наступної змоги використання для огляду даних, а в випадку віддаленого керування комп'ютером, БЕКДОРОМ можуть бути випадкові помилки в коді, які при визвнаі , або підбором констант, або при натиску декількох клавіш можуть давати доступ до деяких даних. Бекдори встановлюються також розробниками для тестування, або з цілю адміністрування. Але вони можуть бутии помічені зловмисниками та використані в особистих цілях.

Проблеми інформаційної безпеки на прикладному рівні. Широке застосування IoT являється результатом інтеграції комп'ютерних технологій, технології зв'язку та різних промишлених областей. Окрім порушення інформаційної безпеки традиційних мереж зв'язку додатки IoT стикаються з додатковими проблемами безпеки на прикладному рівні - при використанні хмарних обчислень, обробки інформації, забезпеченні прав на інтелектуальну власність, захисту приватності та інше.

Деякі дослідження забезпечення інформаційної безпеки IoT. Закордонні спеціалісти приділяють велику увагу науковим та експериментальним дослідженням в забезпеченні інформаційної безпеки IoT. Наприклад в деяких роботах показано, що найбільший риск безпеки можливий на нижньому рівні архітектури – на рівні сприйняття. При цьому відмічається, що деяким загрозам безпеки на інших рівнях архітектури IoT також характерний високий рівень риску.

## 1.2 Основні принципи та характеристики мережі інтернету речей.

IoT базований на трьох принципах.

Перш за все це поширена комунікаційна структура, другим є критична, глобальна ідентифікація кожного пристрою. Третім є можливість надсилання та отримання даних через мережі Інтернет або персональні мережі.

– Найважливішими відмінностями Інтернеті речей від Інтернету який вже наявний є такі:
– Фокус не на людині, а на речах
– Значно велика кількість підключених пристроїв до мережі
– Невисока швидкість передачі даних та значно менші розміри об’єктів
– Фокус не на комунікаціях, а на зчитувані інформації

Концепція покоління NGN припустила можливість комунікації людей безпосередньо, або з використанням комп'ютерів у будь-якому місці в будь-який час. NGN – мультисервісні мережі зв'язку, ядром яких являються опорні IP-мережі, підтримуючі повну або часткову інтеграцію послуг передачі даних та мультімедіа. Реалізує принцип конвергенції послуг електрозв’язку. Концепція інтернету речей включає ще один напрям – комунікація між будь-якими пристроями або речами

Офіційне визначення Інтернету Речей будо приведено у Рекомендації МСЕ-Т Y.2060, відповідно з яким IoT – глобальна інфраструктура інформаційного суспільства, що забезпечує провідні послуги за рахунок організації зв’язку між речами (фізичними чи віртуальними) на основі сумісних інформаційних і комунікаційних технологій, що вже існують або лише розвиваються.

Основним завданням передбачається виконання пристроями певних дій, без жодного втручання людини. Для прикладу, можемо навести, розумний годинник чи навіть ціле місто, адже під’єднашись через мережу до засобів Інтернету, можна надати цим об’єктам нові функції та можливості. Таким

чином, всі пристрої можуть виконувати обробку інформації, проводити аналіз та обмін між собою та, залежно від результатів, приймати рішення і виконувати певні дії.

Однозначно концепція IoT є проривною в сучасному суспільстві. Це відзначає Європейський Союз та Міжнародний союз електрозв'язку, включаючи його у перелік визначних технологій в США, Китаї та інших країнах світу. Незважаючи на те, що концепція вже затверджується, як сформована технологія та набуває популярності серед користувачів, досі продовжується активність в розробці, області стандартизації архітектури, додатків та технічних компонентів.

Окрім згаданого поняття «річ», Міжнародний союз електрозв'язку використовує поняття «прилад» (device) – частина обладнання, функціями якого є комунікація, поєднання можливостей сенсорингу, приведення речей у дію, збору, обробки та зберігання даних. Така система може аналізувати дані навколишнього середовища та виконувати різні дії в залежності від отриманої інформації.

Прикладом впровадження такої концепції є система «розумний будинок» або «розумна ферма». Ця система аналізує дані навколишнього середовища і в залежності від показників регулює температуру в приміщенні. У зимовий період регулюються інтенсивність опалення, а в разі спекотної погоди будинок має механізми відкривання і закривання вікон, завдяки чому провітрюється будинок, і все це відбувається без втручання людини.

Для передачі будь-яких запитів, в мережі Інтернет має конкретну адресу або індифікатор, за допомогою якого можна здійснити зворотній зв'язок з потрібним об'єктом/річчю. У наданні своїх сервісів будь-який вузол є рівноправним, завдяки єдиному протоколу взаємодії, що має інтернет речей.

Раніше існувала проблема пов’язана з протоколом IPv4 – мережні адреси, якого були вичерпані. Саме це перешкоджало у втіленні ідеї Інтернет речей. Проте впровадивши нову версію IPv6 цю проблему було вирішено та дало змогу втілити ідею Інтернету речей в реальність. Ця ідея може вирішувати різні обчислювальні задачі, утворювати місцеву мережу, яка об’єднана функцією чи зоною обслуговування.

Проте, як і кожна унікальна система, Інтернет речей має свої недоліки, що пов’язані з фінансовою стороною реалізації проекту. Оскільки, планується такий продукт, який буде доступний за низькою ціною та з достатньою кількістю об’єктів, варто врахувати усі чинники та проблеми. Тож детальніше розглянемо кожну з них.

– Масштабованість.

Оскільки речі взаємодіють між собою у відкритому середовищі, то і в порівнянні зі звичайним інтернет комп’ютером, Інтернет Речей має набагато ширшу концепцію. Тому варто звернути увагу, на те, що як у малих, так і у масштабних середовищах, важливою роллю має залишатися функціонування таких можливостей, як: встановлення зв’язку та обслуговування. Без ефективної роботи, знаходження нових методів та функцій для масштабованості, розвиток Інтернет речей може значно скоротити свої оберти націлених на високий результат.

– Організованість.

Інтелектуальні інтернет речі не мають керуватися пристроями, які вимагають від користувача їх налаштування і адаптацію в конкретних ситуацій. Мобільні речі, які часто використовуються лише випадково, повинні спонтанно встановлювати зв’язок, при цьому їх можна організовувати та налаштовувати відповідно до їх конкретного середовища.

– Обсяги даних.

Часто у практиці Інтернет речей зустрічається нечасте з’єднання. Це спричинено роботою датчиків мереж, що займаються постійним збором інформації у величезних обсягах на центральних мережевих вузлах або серверах. Окрім нових технологій зберігання, обробки та управління, описане вище явище також потребує значної кількості операційних механізмів.

– Інтерпретація даних.

Для підтримки користувачів «internet things», необхідно приділити увагу інтерпретації місцевого контексту, що визначений датчиками мереж. Для отримання прибутку від неоднорідних даних, які будуть створені, необхідно робити загальні висновки з даних по інтерпретованому сенсору.

– Оперативна сумісність.

Для полегшення зв’язку та кращій співпраці різних інтелектуальних об’єктів необхідні стандартизовані дані. Оскільки кожен тип інтелектуальних об'єктів поділяється на різні можливості: інформаційні, оброблювальні та комунікаційні. Таким чином і працювати кожен тип буде по своєму в різних умовах, таких як доступність енергії та вимоги пропускної здатності зв'язку.

– Автоматизоване виявлення.

Автоматична ідентифікація послуг речей повинна існувати у динамічних середовищах, проте для цього потрібні семантичні засоби для кращого опису їхньої функціональності.

– Складність програмного забезпечення

Складність програмного забезпечення полягає у недостатності ширшої інфраструктури програмного забезпечення в мережі. Розширення

інфраструктури необхідне і на фонових серверах, де відбувається управління динамічними об’єктами та надання послуги для їх підтримки.

– Безпека та конфіденційність.

Пристрої IoT часто мають недоліки безпеки. Їх прошивка, тобто вбудоване програмне забезпечення, може містити старі та вже добре відомі уразливості під час чи після потрапляння на ринок. Ними можуть скористатися кіберзлочинці для проникнення в корпоративну мережу. На додачу з’являються нові вимоги для Інтернету речей, а саме: конфіденційність повідомлень, їх цілісність, автентичність та надійність посередників. З’являється необхідність у дозволу доступу до певних послуг, використання інтелектуальних об’єктів під час різних операцій, що повинні бути у захисті від конкурентів.

– Толерантність до помилок.

Оскільки об’єкти Інтернет речей є динамічними та мобільними, з’являються проблеми у неочікуваних змінах. Тому для швидкої реакції, Інтернет речей повинен мати надійну структуру, для можливості адаптуватися до будь-яких змінених умов.

– Енергопостачання.

Оскільки речі знаходяться у постійному русі, є необхідність у підключенні до автономного джерела енергії. Включаючи фінансові затрати, найвигіднішим варіантом є процесори низької потужності та пристрої зв’язку, які вбудовуються у пристрої. В програмному забезпеченні енергозбереження грає значну роль, тому кожен окремий байт передачі повинен бути обґрунтованим.

– Бездротові зв'язки.

На платформі інтернет речей найпопулярнішими є такі бездротові технології, як: GSM, UMTS, Wi-Fi та Bluetooth, проте з точки зору енергоефективності, вони підходять найгірше. Останні оновлення бездротових технологій – WPAN та ZigBee, які поки знаходяться в розробці, використовують значно меншу потужність, але можуть мати вужчу смугу пропускання.

*Основні характеристики IoT*

– Взаємозв'язок.

IoT має велике охоплення аудиторії, тому будь-що може бути пов'язане з глобальною інформаційною та комунікаційною інфраструктурою.

– Послуги, пов'язані з речами.

В інформаційному та фізичному світі однозначно зміняться такі обмеження, як захист приватності та семантичні відповідності між фізичними об’єктами та об’єктами віртуальної реальності, оскільки IoT постійно їх використовує у своїй сфері діяльності.

– Гетерогенність.

Оскільки пристрої в IoT базуються на різних апаратних платформах та мережах їх називають неоднорідними. Вони можуть з легкістю взаємодіяти з іншими пристроями або сервісними платформами за допомогою різних мереж.

– Динамічні зміни.

У сфері IoT відбуваються постійні динамічні зміни, наприклад: сон та пробудження, підключення та/або відключення. Динаміка спостерігається і в контексті пристроїв, включаючи місцезнаходження та швидкість. До цих параметрів, включається і кількість пристроїв, яка постійно зростає.

– Величезна шкала.

Існує велика кількість пристроїв, що потребують постійного контролю та потребу в керуванні. Взаємодія один з одним є на порядок більшою, ніж у пристроїв, підключених до сучасного Інтернету. Більш критичним буде управління отриманими даними та їх інтерпретація для застосування. Така робота стосується семантики даних, а також ефективної обробки даних.

– Безпека.

Безпека включає в себе збереження особистих даних, а також безпека фізичного стану речей. Захист кінцевих точок, мереж та даних, що переміщуються між ними, означає створення адаптивної парадигми безпеки.

Способи забезпечення інформаційної безпеки інтернета речей

Відповідно до багаторівневої архітектури IoT-систем, можна виділити наступні 3 області в яких потрібно організувати інформаційну безпеку :

– Смарт-пристрої – Датчики, сенсори та інші прибори котрі збирають інформацію з обладнання та відправляють її в хмарне сховище, вертаючи сигнали управління для зміни стану інформації.
– Мережеві шлюзи та канали передачі даних (провідні та безпровідні протоколи)
– Програмні IoT-платформи – хмарні Big Data сервіси обробки інформації та їх зберігання

Для всіх цих компонентів в часності і в цілому для IoT-системи актуальні наступні міри забезпечення кібербезпеки:

а) Організаційні правила:

1) Створення та впровадження єдиної політики інформаційної безпеки підприємства з урахуванням усіх додатків та систем IIoT

2) Розробка правил безпеки використання IoT-присроїв и мереж

3) Удосконалення законодавчого забезпечення недоторканності особистого життя та секретів підприємства

4) Державна та приватна стандартизація та сертефікація пристроїв, каналів передачі даних, сховищ інформації та прикладного ПЗ по обробці та аналізу великих даних

б) Технічні засоби захисту даних від втрати даних, втрати та перехвата управління:

1) Шифрування та інші криптографічні методи. Персоналізація IoT-пристроїв с використанням унікальних ідентифікаторів, MAC-адрес, ключів та сертифікатів, забезпечуючих достатньо високий рівень кібербезпеки без додаткових затрат

2) Гнучкі політики управління доступом с багатофакторними авторизаціями

3) Резервування, реплікація, організація захищеного периметру та інші засоби інформаційної безпеки для великих даних

Способи захисту інформації :

Перешкоди – створення на шляху загрози перешкод, прохід через яку пов’язано з виникненням складностей для зловмисника або дестабілізуючого фактору

Управління – управляючі дії на елементи які захищаються системою

Маскування – дії над захищеною системою або інформацією, які приводять до їх перетворення, які роблять їх недоступними для зловмисника

Регламентація – розробка та регламентація комплексу дій, створюючі такі умови розробки, які істотно ускладнюють можливість атак зловмисника або дії інших дестабілізуючих факторів.

Примус - метод заключається в створенні умов, при яких еористувачі та персонал повинні виконувати умови обробки інформації під загрозою відповідальності.

Спонукання – метод заключається в створенні умов, при яких користувач та персонал виконують умови обробки інформації по морально-етичним ат психологічним спонуканням.

Засоби захисту інформації:

– Фізичні засоби – механічні, електричні, електромеханічні пристрої та системи, які функціонують автономно, створюючи різного роду перешкоди на шлясі дестабілізуючих факторів.

– Апаратні засоби – різні електронні та електромеханічні пристрої, які схемно інтегровані в апаратуру системи обробки інформації спеціально для вирішення задач захисту інформації.

– Програмні засоби – спеціальні пакети програм або окремі програми, інтегровані в склад програмного забезпечення з ціллю вирішення задач захисту інформації.

– Організаційні засоби – організаційно-технічні дії, які спеціально створенні для функціонування системи з ціллю вирішення задач захисту інформації.

– Законодавчі засоби – нормативно-правові акти, за допомогою яких регламентуються права та обов'язки, а також встановлюється відповідальність всіх особистостей та підрозділів, які мають відношення до функціонування системи, за порушення правил обробки інформації.

– Психологічні засоби – моральні норма та етичні правила, які склалися в спульноті або даному колективі, які допомогають в захисті інформації, а не виконання їх прирівнюється до порушенню правил в еколективі або спільноті.

Методом поліпшення захисту інформації в мережі інтернету речей на рівні програмного забезпечення є перевірка, та кодування різного роду.

Симетричні криптографічні системи – методи шифрування, в яких для кодування та декодування використовується один крипто-ключ. З двох сторін ключі алгоритмів потребують зберігання в таємниці. До того як винайшли схеми асиметричних шифрувань усі системи користувались симетричним шифруванням.

Метод кодування та декодування даних розповсюджено застосовуються в комп'ютерних технологіях в моделях приховування приватних і комерційних даних від не санкціонованого застосування не бажаними особами. Основним принципом в алгоритмах є умова, що фізична особа яка приймає повідомлення, знає алгоритм шифрування та дешифрування, а також ключ до розкриття повідомлення, без якого повідомлення є всього лише набір довільних символів, що не несуть корисної інформації.

В залежності від типу ключа криптоалгоритми проводять зміну над невеликими блоками даних отже аби була виключена вірогідність зчитування оригіналу інформації за відсутності секретного ключа.

Найефективнішими системами криптографічного захисту інформації є аси метричні криптосистеми, звані також криптосистемами з відкритим ключем. У подібних системах для шифрування даних застосовується один ключ, а для деш ифруваня - інший ключ. Перший ключ є публічним та може бути переданий для використання всіма юзерами системи, які шифрують дані. Розшифрування інформації за допомогою публічного ключа неможлива [2].

Секретний ключ використовується одержавучем для розшифровки зашифрованої інформації. З ключа зашифровки ми не можемо визначити ключ розшифровки.

Кс повинно обсислюватися за невідомим рівнянням для розкриття секретного ключа за відкритим ключом.

Криптосистеми на основі еліптичної кривої. Останні досягнення теорії обислюваної складності показали, що загальна проблема логарифмування в кінцевих полях, не може являтися достатньо стійким фундаментом. Найбільш ефективні на сьогоднішній день алгоритми дискретного логарифмування мають вже не експоненціальну, а субекспоненціальну часову складність. Це алгоритми “indexcalculus”, які використовують факторну базу, до числа яких відноситься алгоритм Адлемана, та декілька версій “COS”.[3]

Ведуться роботи по розвитку ефективності цих алгоритмів. Ряд успішних атак базуються на складності дискретного логарифмування в кінцевих полях, привів до того, що стандарти були переведені на еліптичні криві. Схеми ЕЦП при всьому залишилися такими самими, але в якості чисел, якими вони оперують тепер використовується не елементи кінцевого поля, а еліптичні числа – рішення рівнянь еліптичних кривих над кінцевими полями.

Роль операції приведення числа в степінь в кінцевому полі в оновлених стандартах виконує операція взяття кратної точки еліптичної кривої – множення точки на ціле число. Правильний вибір типа еліптичної кривої дозволяє ускалднити задачу взлому схеми ЕЦП і зменшити робочий розмір блоків даних.

Старий стандарт оперує 1024-бітовими блоками, а новий, базований на еліптичних кривих, - 256 бітовими, і при цьому має більшу стійкість. Криптосистеми на основі еліптичної кривої отримують все більше розповсюдження скоріш як альтернатива, а не заміна системам на основі RSA. Вони мають деякі переваги, особливо при використанні в пристроях з малопотужними процесами та невеликою памятю

### 1.3 Вибір прототипу на основі існуючих рішень

SOAP API

Веб-сервіс дає змогу взаємодії з сервером чітко структурованими повідомленнями. Справа в тому, що веб-сервіс не приймає будь-які дані. На будь-яке повідомлення, яке не відповідає правилам, веб-сервіс відправляє помилку. Помилка буде повернута в форматі XML з чіткою структурою.

WSDL (Web Services Description Language). Правила за якими створюються повідомлення для веб-сервісу описуються також за допомогою XML і також мають чітку структуру  тобто, якщо сервіс надає можливість виклику якогось методу, він має надати клієнтам можливість надати дізнатися які параметри для даного метода використовуються, якщо веб сервіс очікує рядок для методу в якості параметру і рядок повинен мати ім'я, то в описі веб-сервісу ці правила повинні бути вказані.

В якості параметрів, можуть передаватися не тільки прості типи та об'єкти, але і об'єкти, колекції об'єктів. Опис об'єкту зводиться до опису

кожного сладаючої чатсини об'єкту. Якщо об'єкт складається з декількох поле, отже кожне поле описується, який його тип та назва та інші описуючі методи. Загалом простими можуть бути якісь специфічні, важливо щоб клієнт міг зрозуміти їх зміст.

Для клієнта достатньо знати URL веб сервісу, WSDL завжди буде поруч по якому можна отримати представлення про методи та їх параметрах, які надає цей сервіс.

JSON-RPC

**JSON-RPC** – сервіс віддаленого виклику процедур, який використовую JSON в якості формату даних. Даний сервіс в багато чому схожий на XML-RPC, специфікація визначає декілька типів даних і правил їх обробки. JSON-RPC підтримує повідомлення і багаточислені виклики, так як інформація, яка приходить на сервер потребує відповіді.

JSON-RPC проводить відправку запросів на сервер, реалізуючий віддалений протокол. Як клієнт в більшості випадків виступає програма, яка викликає метод на віддалений протокол. Вхідні параметри передаються в якості масива або об’єкта, в деяких віпадках можлива передача вихідних даних. Процес проходит за допомогою відправки запитів на віддалений сервер через http або tcp/ip сокет. В випадку http заголовок Content-Type визначається як application/json

Особоливості:

1) Стандартизований та має велику кількість бібліотек для різних платформ.
2) Простий парсинг, краще аналога XML-RPC.
3) Обробка помилок являється влаштованою функцією.

4) Підтримується черга викликів.
5) Єдина точка входу для API.
6) Підтримуються оригінальні параметри при виклиці методів.

XML-RPC

**XML-RPC** — стандартизований виклик віддалених процедур, які користуються XML для кодування своєї інформації та HTTP в якості транспортного механізму. Який є предком SOAP, відрізняється виключною простотою в застосувані.

XML-RPC, як і будь-який інший интерфейс Remote Procedure Call (RPC), визначає набір стандартних типів даних та команд, котрі програміст може використовувати для доступа до функціональності іншої програми, яка знаходиться на іншому комп'ютері в мережі.

REST API

REST API має наувазі прості правила:

– Кожен URL являється ресурсом
– При звернені до ресурсів методом GET вертається опис цього ресурсу
– Метод POST додає новий ресурс
– Метод PUT змінює ресурс
– Метод DELETE видаляє ресурс

Ці правила надають простий CRUD інтерфейс для інших додатків, взаємодія з якими проходить через протокол HTTP.

Відповідність CRUD операцій та HTTP методів:

- **CREATE** - POST
- **READ** - GET
- **UPDATE** - PUT
- **DELETE** - DELETE

REST API інтерфейс дуже зручний для міжпрограмної взаємодії, наприклад мобільний додаток може виступати в ролі клієнта, котрий маніпулює даними через REST.

Таблиця 1.1

Порівняльна таблиця програмного забезпечення

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Призначення | Простий протокол доступу до об'єктів | Представницький стан передачі | Протокол віддаленого виклику процедур | Протокол віддаленого виклику процедур |
| Концепт | Стандартизований протокол із заздалегідь визначеними правилами. | Архітектурний стиль із без вказівок та з рекомендаціями. | Простий протокол визначаючий тільки декілька типів даних та команд. | Протокол виклику віддалених процесів, використовуючих XML для кодування своих повідомлень HTTP в якості транспортного механізму |
| Підхід | Орієнтований на функціонал | Орієнтований на дані | Орієнтований на дані | Орієнтований на дані |

Продовження таблиці 1.1

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Залежність від стану | За замовчуванням немає стану, але можливо зробити SOAP API зі станами | Без стану (без сеансів на стороні сервера). | Без стану (встановлюютьс я сеанси на стороні клієнта). | Без стану (без сеансів на стороні сервера). |
| Кешування | Виклики API не можна кешувати. | Виклики API можуть кешуватися . | Виклики API не можна кешувати. | Виклики API можуть кешуватися. |
| Безпека | WS-безпека з підтримкою SSL. Вбудована відповідність ACID. | Підтримує HTTPS та SSL. | Підтримка рівню безпеки за допомогою TLS | Підтримка рівню безпеки за допомогою TLS, SSL |
| Продуктивніст ь | Потрібна більша пропускна здатність та обчислювальн а потужність. | Потрібно менше ресурсів. | Не потребує великих русурсів | Потребує більших ресурсів для коректної продуктивност і |

Продовження таблиці 1.1

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Формат повідомлень | Тільки XML | Звичайний текст, HTML, XML, JSON, YAML та інші. | Тільки JSON | Тільки XML |
| Протокол передачі | HTTP, SMTP, UDP та інші. | Тільки HTTP | HTTP, TCP/IP сокет | Тільки HTTP |
| Рекомендовано для | Корпоративні програми, додатки високої безпеки, розподілене середовище, фінансові послуги. | Публічні API для веб-служб, мобільних служб, соціальних мереж. | Використовується для с систем з Batch-запитами, Вебсокети, Нотифікації, котрі можна обробляти асинхронно | Корпоративні програми, розподілене серидовище |

Продовження таблиці 1.1

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Переваги | Висока безпека, стандартизованість, розширюваність. | Масштабованість, кращі показники, зручність роботи з браузером, гнучкість. | За одну ітерацію запиту можно виконувати декілька дій на сервері | Облегчення створення бібліотек, котрі автоматично сериалізують /десериалізуют ь об'єкти уровня мови в викликах RPC, тому він имеет деякі незначні переваги при використання |

Таблиця 1.2

Таблиця оцінки порівняння програмного забезпечення

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Призначення | 4 | 3 | 2 | 2 |
| Концепт | 3 | 4 | 2 | 1 |
| Підхід | 3 | 4 | 4 | 4 |
| Залежність від стану | 1 | 4 | 3 | 2 |
| Кешування | 2 | 4 | 2 | 4 |
| Формат повідомлень | 1 | 4 | 1 | 1 |

Продовження таблиці 1.2

| Критерій | SOAP | REST | JSON-RPC | XML-RPC |
|---|---|---|---|---|
| Безпека | 3 | 4 | 2 | 2 |
| Продуктивність | 2 | 4 | 3 | 1 |
| Протокол передачі | 4 | 2 | 3 | 2 |
| Рекомендовано для | 4 | 2 | 3 | 1 |
| Переваги | 3 | 2 | 1 | 4 |

Виходячи з результатів об’єктом мого дослідження та удосконалення обираємо REST API так, як ця технологія являється більш гнучкою, та простою до удосконалення.

### 1.4 Огляд застосування REST API за основу прототипу.

Розглянемо REST API як частину архітектури мережі інтернету речей.

REST API використовуються як проміжне програмне забезпечення, тобто виступає в ролі сервера між клієнтом та сховищем даних.

REST є збіркою правил для стандартизованого зв’язку між клієнтом та сервером.

Що таке RESTful:

Щоб система вважалась сконтрюйовнаою по REST архітектурі необхідно, шоб вона задовільняла наступні критерії.

1) Client-Server. Система повинна бути розподілена на клієнтів та серверів. Розподілення інтерфейсів означає, що наприклад клієнти не

зв’язані напряму з сховищем даних, яке залишається всередені кожного сервера, так щоб покращилась мобільність кода клієнта Сервери не пов’язані з інтерфейсом користувача, отже сервери можуть бути простішими та масштабуючими. Сервери і клієнти можуть бути взаємозамінюючими та розроблятися окремо.

2) Stateless. Сервер на має зберігати інформацію про клієнтів . В запиті повинна передаватися вся необхідна інформація для його обробки і ідентифікація клієнта.

3) Cache. Кожна відповідь має бути відмічена чи є чи потрібно його поміщати в кеш, або ні, для того щоб ми могли попередити повторне використання клієнтами застарілих або некоректних даних в відповідь на подальші запити.

4) Uniform Interface. Єдиний інтерфейс визначає інтерфейс між клієнтом та сервером. Це спрощує та відділяє архітектуру, яка дозволяє кожній частині розвиватися самостійно. Чотири принципа єдиного інтерфейсу:

- Identification of resources (заснований на ресурсах). В REST ресурсом являється усе те чому ми можемо надати ім'я, або ідентифікатор. Наприклад, користувач, зображення, предмет. Кожен ресурс в REST повинен бути ідентифікований за допомогою стабільного ідентифікатору, який не змінюється при зміні стану ресурса. Ідентифікатором в REST являється URI.

- Manipulation of resources through representations. (Маніпуляція над ресурсами через представлення). Представлення в REST використовується для виконання дій над ресурсами. Представлення

ресурса представляє собою поточний або бажаний стан ресурса. Наприклад, якщо ресурсом являється користувач, то представленням може являтися XML або HTML опис цього користувача.

- Self-descriptive messages (само-опис повідомлення). Під само-описом мається наувазі, що запит і відповідь повинні містити в собі всю необхідну інформацію для їх обробки. Не повинні бути додаткові повідомлення або кеш для обробки одного запиту. Іншими словами відсутність стан, сохраняємого між запитами до ресурсів. Це дуже важно для масштабування системи.

- HATEOAS (hypermedia as the engine of application state). Статус ресурсу передається через інформацію, що міститься в body, параметри запиту, заголовки запитів та потребуючий URI (ім'я ресурсу). Це називається гіпермедіа HATEOAS також означає, що, в випадку необхідності силки можуть міститися в тілі відповіді (або заголовках) для підтримки URI.

5) Layered System. В REST допускається разділити систему на ієрархію слоїв але з умовою, що кожен компонент може бачити компоненти тільки безпосередньо наступного слою. Наприклад, якщо ви викликаєте службу PayPal, а він в свою чергу викликає службу Visa, про виклик служби Visa нічого не повинні знати.

6) Code-On-Demand (опціонально). В REST дозволяється загрузка та виконання коду якщо програми на стороні клієнта.

Сервери можуть тимчасово розширяти або кастомізувати функціонал клієнта, передаючи йому логіку, яку він може виконувати. Наприклад, це можуть быть скомпільовані Java-апплети або клієнтські скрипти на Javascript

**Висновки:**

1) Розглянуто проблеми захисту інформації в мережі інтернету речей. Проведений огляд допоміг визначити проблеми, які присутні в мережі інтернету речей, зокрема проблема захисту інформації даних.

2) Проаналізовано основні поняття та концепції інтернету речей. Розглянуто три основні принципи в мережі інтернету речей та основні концепції сучасної розробки систем інтернету речей.

3) Проведено огляд та аналіз існуючих рішень підвищення рівню захисту інформації. Проведено порівняльний аналіз проміжного ПЗ, за допомогою аналізу було обрано прототип для запропонованого методу.

4) Проведено детальний огляд обраного методу як прототипу, розглянуто основні принципи використання проміжного ПЗ.

# РОЗДІЛ 2.

# УДОСКОНАЛЕНИЙ МЕТОД ЗАХИСТУ ІНФОРМАЦІЇ ЗА РАХУНОК ВПРОВАДЖЕННЯ REST API ЯК ПРОМІЖНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ЕЛІПТИЧНОЇ КРИПТОГРАФІЇ

## 2.1 Метод застосування REST API в мережі IoT

REST API дозволяють відкрити підключений пристрій користувачам у програмі безпечним способом. RESTful API широко використовуються в сучасній мережі. Передача даних зазвичай здійснюється за допомогою JSON або XML через HTTP. Це гарна модель для неоднорідних систем. REST API робить інформацію про пристрій легкодоступною. Вони можуть стандартизувати спосіб створення, читання, оновлення та видалення даних. Усі ці операції включені до REST-запиту [16]. API REST дозволяють делегувати та керувати авторизацією. API може автентифікуватись на сервері і сервер можуть аутентифікувати API, щоб запобігти атаки в середині [5].

На рис. 2.1 представлена загальна картина ролі проміжного програмного забезпечення в IoT. У загальному сенсі маємо чотири категорії основних компонентів системи IoT - датчики, локальна мережа, яка може включати шлюз, проміжне програмне забезпечення, хмарне сховище [1].

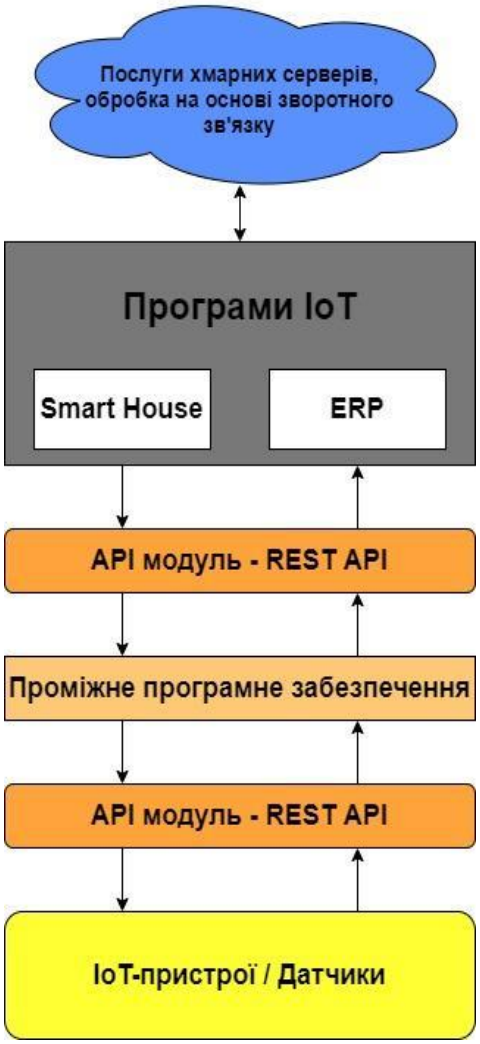


Рис. 2.1 Модель архітектури Проміжного програмного забезпечення в мережі IoT

Доступно багато протоколів авторизації. Наприклад, OAuth - це відкритий протокол авторизації, який може надати доступ до ресурсів через програмне забезпечення, використовуючи ім'я користувача, пароль та токени [7].

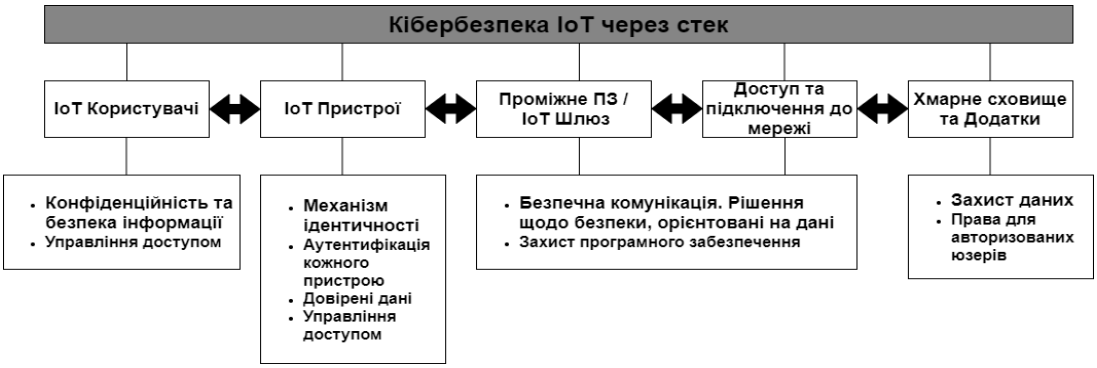


Рис. 2.2 Модель захисту для системи інтернету речей

Протокол обмеженого застосування:

– Частина IPV6 для обмежених ресурсних середовищ

– Інтерфейс API REST, орієнтований на IoT, відображається на ефективний бінарний протокол

– Кінцеві пристрої можуть мати як клієнтську, так і серверну ролі, також можуть безпосередньо спілкуватися

– Представлення ресурсів, наприклад 'application/json' , узгоджується як у протоколі http

– Дотримуйтесь опції для тривалої реакції потоку подій на GET, асинхронні оновлення

– Веб-зв'язок стандартного формату основного посилання IETF

– Виявлення ресурсів за допомогою /well-known/core або зовнішнього каталогу ресурсів

HTTP/REST:

– HTTP та REST корисні для більшості веб-додатків, служб та шлюзів

– Використовує веб-виклики, веб-сокети, HTTP PUT для асинхронних оновлень

– Посилання в JSON, Link-format, Hypercat, пов'язані між собою дані

- Веб-посилання для IoT

– Інкапсуляція об'єктів: об'єкти, що описують себе, мають посилання, пов'язані з кінцевими точками, наприклад /well- known/core

– Каталог ресурсів: дерикторії та каталоги посилань, що зберігаються у відомих місцях, що вказують на ресурси та інші каталоги

– Посилання можуть бути зареєстровані в Каталозі ресурсів, коли сенсори під'єднані до мережі

– Сканер може знаходити посилання за адресами /well-known/core та заповнювати каталоги

– Прикладне програмне забезпечення може виявляти ресурси за атрибутом, наприклад запит, використовуючи type = 'temperature', units = 'celsius', location = 'outdoors'

– Система повертає набір посилань, що відповідають атрибутам запиту

## 2.2 Модифікований метод захисту інформації

Для модифікації методу який базується на REST API для підвищення рівню безпеки інформації в мережі, я пропоную використання криптографічного алгоритму еліптичних кривих.

**Протокол Діффі-Хеллмана на еліптичних кривих** — криптографічний протокол, який дозволяє обом сторонам, що мають сполучення відкритий/закритий ключ на еліптичній криптографії, отримати спільний секретний ключ, використовуючи незахищений від загроз шлях зв'язку. Цей секретний ключ може бути застосований як для шифрування подальшого обміну, так і для формування нового ключа, який потім може використовуватися для подальшого обміну інформацією за допомогою

методів симетричного шифрування. Це варіант протоколів з використанням еліптичних кривих [15].

**Опис алгоритму**: Нехай абоненти: Клієнт та Сервер. Наприклад, Клієнт створює спільний секретний ключ з Сервером, але доступний між ними канал може бути прослуховуваним зловмисниками. Спершу повинен узгодитися набір параметрів ( поля характеристики, загальні випадки ). Відповідно у кожної зі сторін повині бути ключі, що складаються з секретного ключа , та публічного ключа.

Більша кількість протоколів, які основані на ECDH, використовують функціонал процесу формування ключів для отримання значення симетричного ключа . З цієї інформації, зв'язаної зі своїм секретним ключом, повідомляється Клієнт тільки про свій публічний ключ. Отже окрім клієнту ніхто не зможе обрахувати її секретний ключ, окрім пристрою який має права вирішити задачу дискретного логарифмування на еліптичних кривих. Секретний ключ Серверу також захищений. Ніхто окрім клієнту або серверу не може обчислити їх спільний секретний ключ, окрім учасника який має права вирішити проблему Діффі — Геллмана [8].

Публічні ключі: статичними або ефемерні. Ефемерні ключі застосовуються лише на конкретний проміжок часу та не автентифікують абонента, отже, якщо потребується авторизація, верифікація автентичності має біти отриманим іншим методом.

Якщо Клієнт або Сервер мають спільний статичний ключ, вразливість атаки посередника не враховується, не може забезпечуватися ні прямий секрет, ні стійкість підміни під час декриментації ключів, також інші параметри стійкості до вразливостей. Користувачі секретних ключів повинні перевіряти інший публічний ключ і застосовувати функцію процесу формування ключів на

спільний секретний ключ, для того щоб виключити витоків інформації про статично секретний ключ [14].

Для кодування з іншими параметрами часто застосовують протокол MQV(Menezes–Qu–Vanstone).

При застосуванні спільного секретного ключу, часто бажано кешувати секретну інформації, для того щоб позбутися від атак, що з'явилися після використання протоколу.

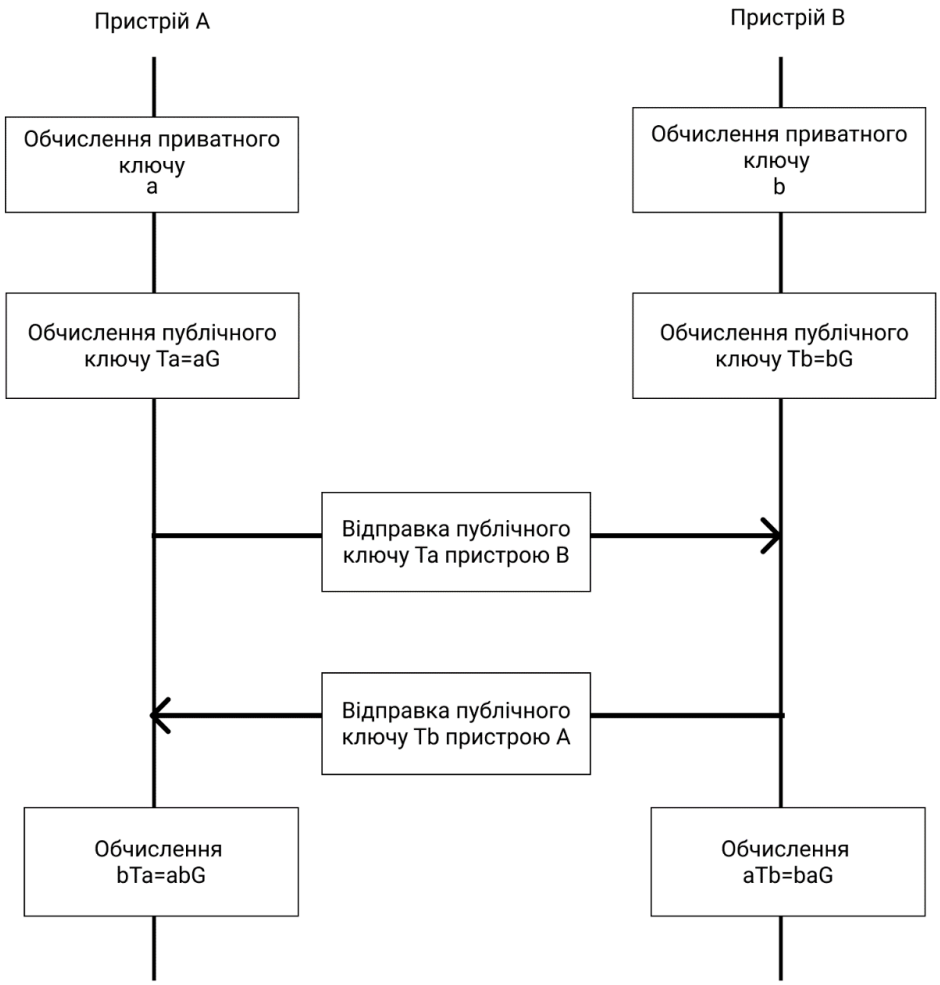


Рис. 2.3 Блок-схема роботи алгоритму еліптичної криптографії

Для реалізації на серверній частині цього алогоритму використано бібліотеку “crypto” для формування публічного та секретного ключу.

Для здобуття цілі підвищення захисту в інформації в мережі інтернету речей я пропоную такий алгоритм:

1) Розробка REST API та дерева URL для запитів з клієнту.
2) Розробка програмного забезпечення для клієнту.
3) Розробити логіку запитів.
4) Формуємо на сервері публічний ключ пристрою за допомогою Протоколу Діффі-Хе́ллмана на еліптичних кривих використовуючи будь який тип кривих та кодуємо його в типі base64.
5) При відправленні інформації на сервер спочатку мі повинні запит на сервер для отримання Публічного ключу який зберігається на сервері.
6) В тілі запиту ми маємо відправляти ідентифікатор пристрою.
7) Запит оброблюється на сервері, формується тіло відповіді серверу на запит, в тілі відповіді в нас передається публічний ключ в зашифрованому виді. Шифрується цей ключ за допомогою симетричного алгоритму блочного шифрування «AES256» в якому ключом до шифру виступає ідентифікатор. Таким чином ми добиваємося того, що при кожному запиті на сервер вертатися в нас буде зашифрований публічний ключ серверу і кожен раз він буде різний.
8) На стороні клієнту за допомогою ключу шифрування ми розшифровуємо публічний ключ серверу.
9) На стороні клієнту формуємо публічний ключ для даного запиту за допомогою Протоколу Діффі-Хе́ллмана на еліптичних кривих використовуючи будь який тип кривих та кодуємо його в типі base64.

10) Наступним кроком є формування секретного ключу пристрою для даного запиту. Формуватися секретний ключ буде на основі публічного ключу сервера, який ми отримали раніше.

11) Для коректної передачі даних ми повинні їх конвертувати до типу даних string.

12) Кодуємо дані які ми хочемо передати за допомогою алгоритму “AES256” в якому ключем шифрування виступає секретний ключ клієнта який ми сформували раніше.

13) Формуємо змінну яку ми будемо передавати в тілі запиту вона буде формуватися з рядку в якому буде в одному рядку публічний ключ, та в якості додаткових розрядів буде виступати публічний ключ зашифрований за допомогою AES256 в якості ключа до якого буде виступати публічний ключ серверу.

14) Робимо запит в тілі якого буде передаватися об’єкт з двома полями Поле 1 це змінна яку ми формували на кроці 13 Поле 2 зашифрованні дані які ми формували на кроці 12.

15) На сервері Розшифровуться Поле 1 та ми отримуємо публічний ключ клієнта.

16) На основі отриманого публічного ключу клієнту ми формуємо секретний ключ серверу, що формує між клієнтом та сервером сеансову пару ключів в ході чого секретний ключ сформований на клієнті буде таким же як і сформований секретний ключ на сервері.

17) Отримавши секретний ключ серверу ми розшифровуємо передані дані в зашифрованому виді.

18) Конвертуємо отриманий рядок до першопочаткового формату.


Рис. 2.4 Блок-схема запропоновнаого методу

В ході цього алгоритму ми отримуємо захищену систему передачі, де для кожного запиту всі дані які передаються будуть зашифровані та зашифровані дані не будуть повторюватися навіть якщо тіло запитів біде абсолютно однаковим. Завдяки використанню алгоритму еліптичної криптографії під час процесу передачі даних у нас утвоюється між сервером та клієнтом сеансова пара, чрезе те що в нас секретний ключ пристрою утвоюється на основі публічного ключу іншого пристрою, секретні ключі обох пристроїв під час одного запросу є однаковими.

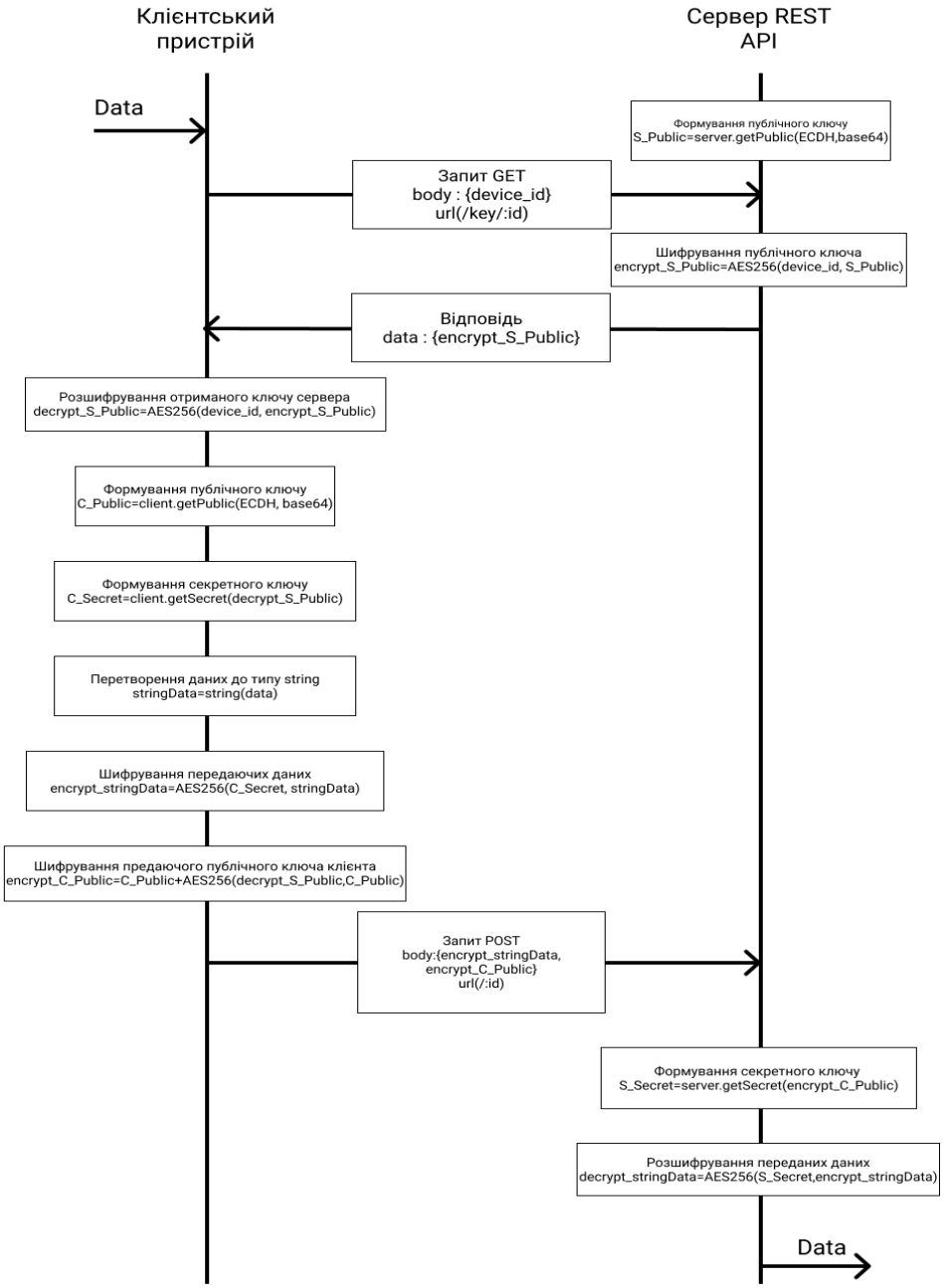


Рис. 2.5 Блок-схема роботи алгоритму запропонованого методу

При виконані запросу на сервер ми виводимо в таблицю дані які ми отримуємо в тілі запиту та розшифровані дані в іншу таблицю.

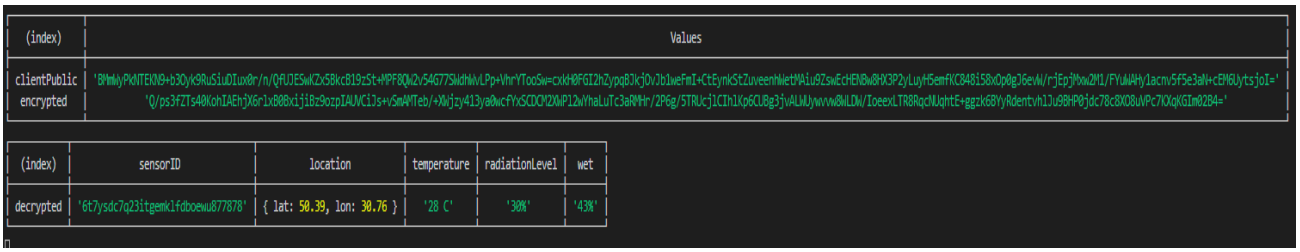


Рис. 2.6 Зображення прийнятих та розшифрованих даних на сервері

**Висновки:**

1) Розглянуто використання взятого за прототип метод захисту проміжного ПЗ в мережі інтернету речей, приведено детальну схему використання в мержі інтернету речей, розглянуто переваги та недоліки його використання, обрано, що на основі представлених недоліків буд проводитися модифікування.

2) Запропоновано модифікований метод захисту інформації в мережі інтернету речей, який базується на REST API та на методі Еліптично-кривої криптографії. Для даного методу приведено алгоритм та блок схему роботи даного методу.

# РОЗДІЛ 3.

# ПРОВЕДЕННЯ НАТУРНОГО МОДЕЛЮВАННЯ МЕТОДУ ТА НАДАННЯ АНАЛІТИЧНОЇ ОЦІНКИ ЗАПРОПОНОВАНОГО МЕТОДУ

## 3.1 Натурне моделювання

Моделювання за часом

– Метод передачі даних без шифрування

При передачі даних без шифрування алгоритм складається з одного кроку, це відправка запиту по визначеному URL

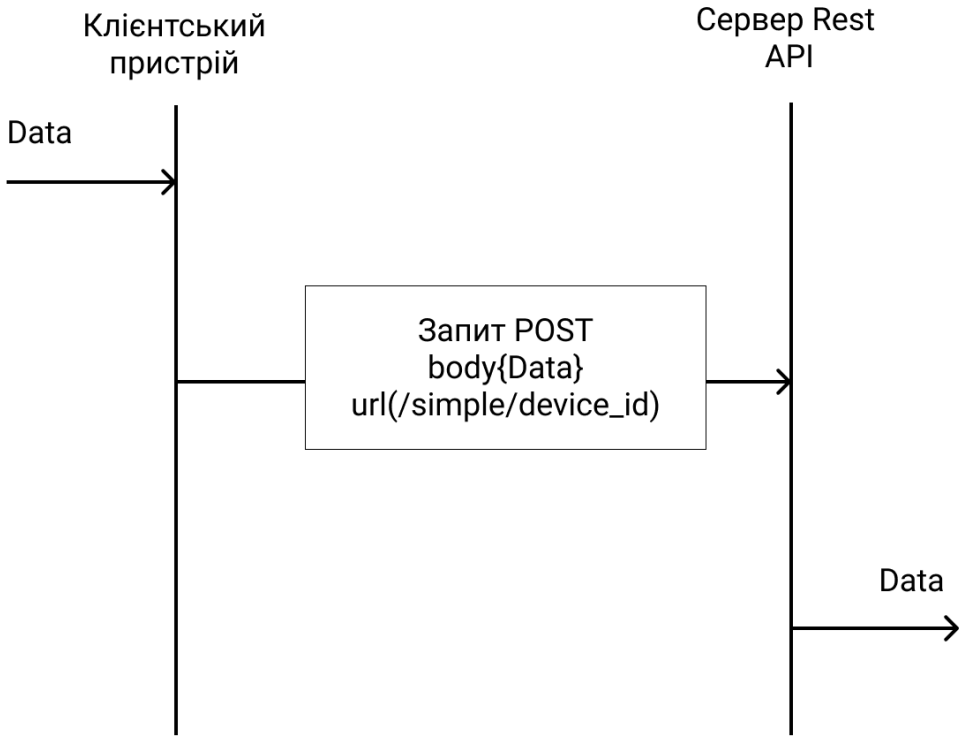


Рис. 3.1 Блок-схема методу передачі інформації без шифрування

– Метод шифрування на основі AES256

Алгоритм:

1) Отримані дані перетворюємо до типу даних string для полегчення процессу шифрування даних
2) Наступним кроком ми шифруємо дані за допомогою алгоритму AES256 де ключом шифрування виступає унікальний ідентифікатор
3) Робимо запит POST на сервер де тілом запиту виступає зашифрований рядок даних за URL(/another/device_id)
4) Отримуємо ключ розшифрування (device_id) з параметрів запиту, який зберігається у полі id
5) Розшифровуємо дані, за допомогою алгоритму AES256 де ключом виступає device_id
6) Перетворюємо отримані дані до першопочаткового формату

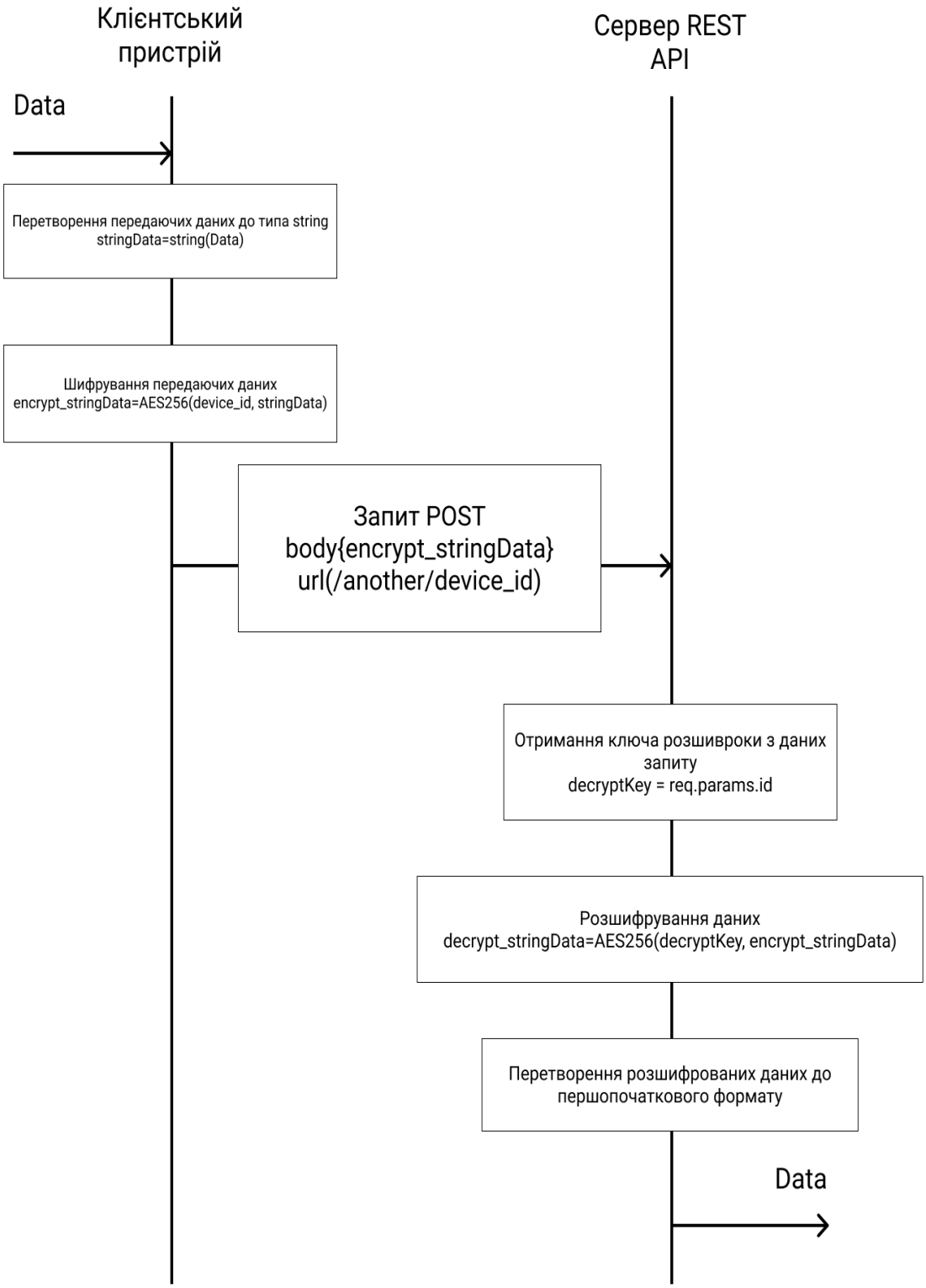


Рис. 3.2 Блок-схема методу передачі інформації на основі AES256

Таблиця 3.1

Результати експерименту за часом

| | Запропонований метод | Метод без шифрування | Метод шифрування AES256 |
|---|---|---|---|
| 1 | 18,71 | 0,006100002676 | 5,489599998 |
| 2 | 15,010799 | 0,01450100169 | 7,362701 |
| 3 | 15,1568 | 0,01000000065 | 6,0143 |
| 4 | 17,743399 | 0,007400002331 | 5,163898999 |
| 5 | 9,296999999 | 0,007801002823 | 8,6115 |
| 6 | 10,82220001 | 0,002999998629 | 6,643200004 |
| 7 | 14,927801 | 0,01090000197 | 3,681800001 |
| 8 | 8,488200004 | 0,003499999642 | 5,607600002 |
| 9 | 12,8772 | 0,002799998969 | 4,321499996 |
| 10 | 8,1395 | 0,01590000361 | 6,1549 |
| 11 | 10,6682 | 0,01229999866 | 4,411201001 |
| 12 | 14,2457 | 0,003600999713 | 7,287799998 |
| 13 | 6,639701003 | 0,003100000322 | 13,0737 |
| 14 | 8,439598997 | 0,00989999529 | 4,683600003 |
| 15 | 7,887300002 | 0,006500001997 | 3,9133 |
| 16 | 7,331600003 | 0,0251000002 | 3,100099998 |
| 17 | 12,0155 | 0,002900000662 | 4,7441 |
| 18 | 7,8438 | 0,002601001412 | 5,941499999 |
| 19 | 8,104500002 | 0,01330000162 | 5,407100001 |
| 20 | 7,087000003 | 0,002500999719 | 4,484399998 |
| 21 | 7,317600002 | 0,008698995225 | 4,413099998 |
| 22 | 9,009601 | 0,008600000292 | 3,804601003 |

Продовження таблиці 3.1

| | Запропонований метод | Метод без шифрування | Метод шифрування AES256 |
|---|---|---|---|
| 23 | 9,196599998 | 0,0250000013 | 4,602801001 |
| 24 | 5,805200002 | 0,003200002015 | 4,879600001 |
| 25 | 8,319199997 | 0,002500999719 | 5,187198998 |
| 26 | 8,779999999 | 0,002498999238 | 5,263398994 |
| 27 | 6,947698998 | 0,007299996912 | 4,268400001 |
| 28 | 7,098999998 | 0,003400001675 | 7,187499998 |
| 29 | 10,1271 | 0,004200000316 | 5,5028 |
| 30 | 6,498200998 | 0,005200002342 | 5,326401003 |

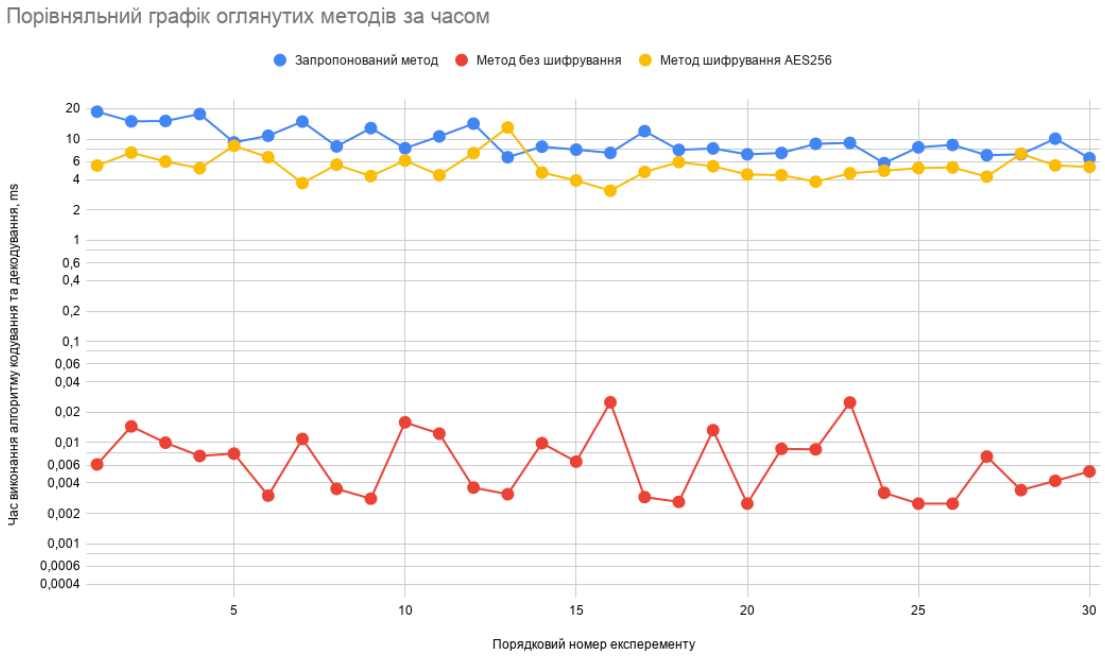


Рис. 3.3 Порівняльна діаграма часу виконання алгоритмів

Для проведення тестування та порівняння оглянутих методів використано двохланкову архітектуру клієнт-сервер, де сервером

виступає REST API який налаштований на роботу з кожним із методів, клієнтом виступає веб-додаток на основі React.js який іммітує поведінку відправки даних від датчика до сервера з викростанням бібліотек для шифрування.

Для урахування часу виконання алгоритму враховано лише час виконання усіх процессів шифрування на клієнті та дешифрування, до разрахунків не береться час виконання запиту так як це напряму залежить від швидкості інтернет з'єднання та фізичного розташування сервера та клієнтського пристрою.

При моделювані клієнту враховується алгоритм шифрування даних час виконання цього алгоритму напряму залежить від продуктивності Browser Engine на реальному пристрою аналогічно за ресурсами пристрою такими як показники процесору.

При розрахунку часу виконання алгоритму декодування на сервері час виконання цього алгоритму залежить фід фізичних показників пристрою на якому знаходиться сервер в ході цього експеременту фізичним сервером виступав ПК за процессором Intel® Core™ i5-8365U (1,60 GHz) та відеокартою NVIDIA GeForce MX150, RAM 8Gb.

Тестовим об'єктом відправки виступає об'єт з об'ємом пам'яті 648 байт формату JSON з такими полями як sensorId, location, temperature, radiationlevel, wet. Схема об'єкта така:

```
{
    "sensorId": [String],
    "temperature":[String],
    "radiationLevel":[String],
    "wet":[String],
```

```
“location”:{
“lat”:[Number],
“lon”:[Number]
        }
}
```

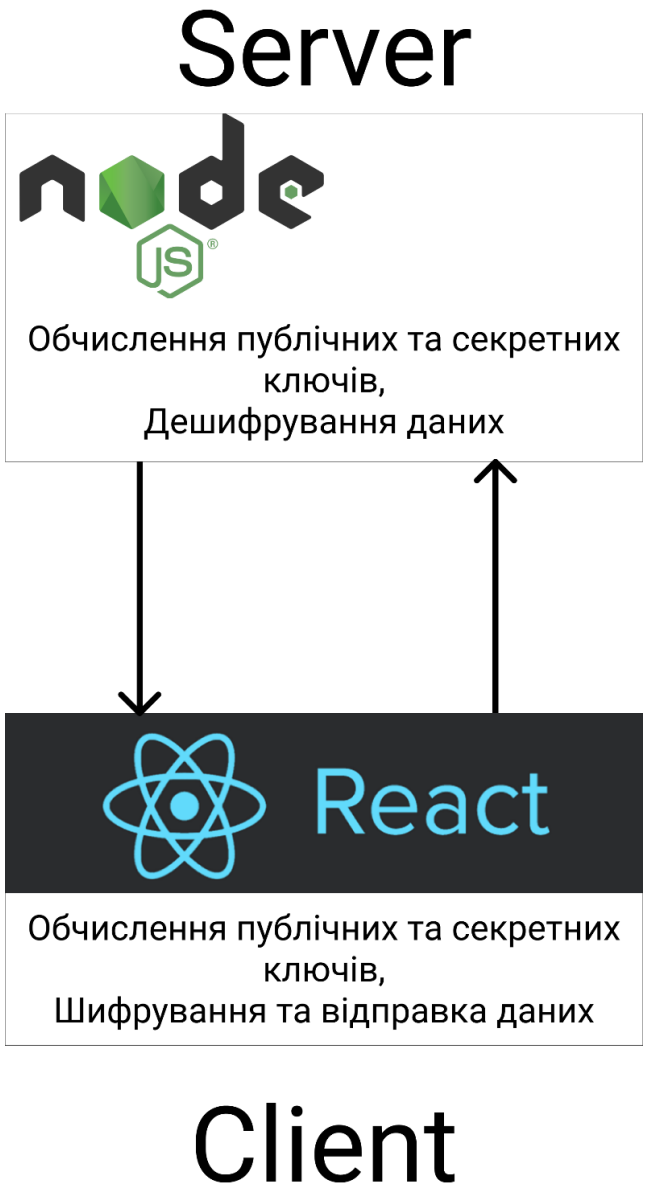


Рис. 3.4 Модель клієнт серверного зв’язку при моделюванні

Для виконання експерименту був модифіковані методі та додані елементи для обчислення виконання алгоритму

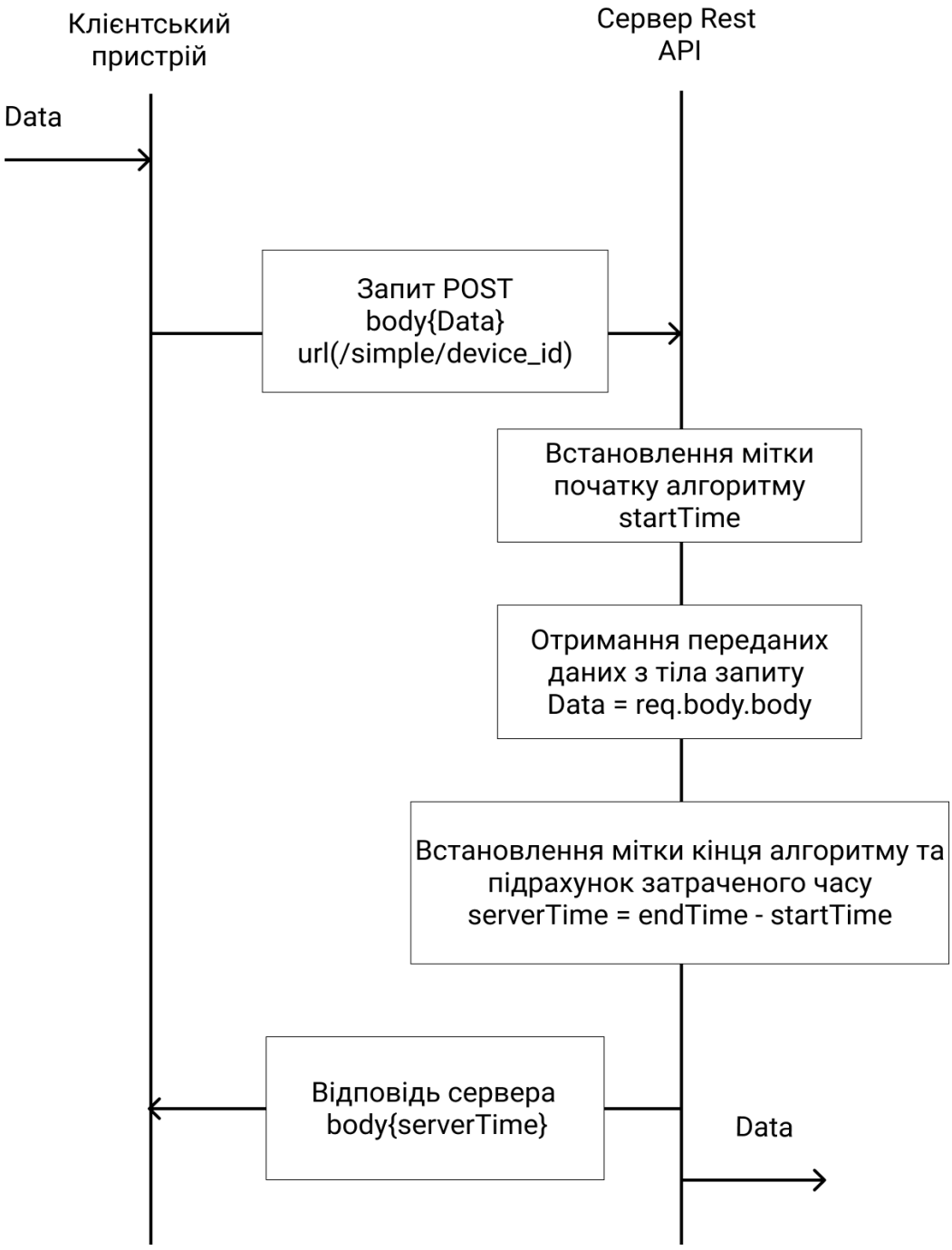


Рис. 3.5 Блок-схема методу передачі без шифрування для обчислення часу виконання алгоритму

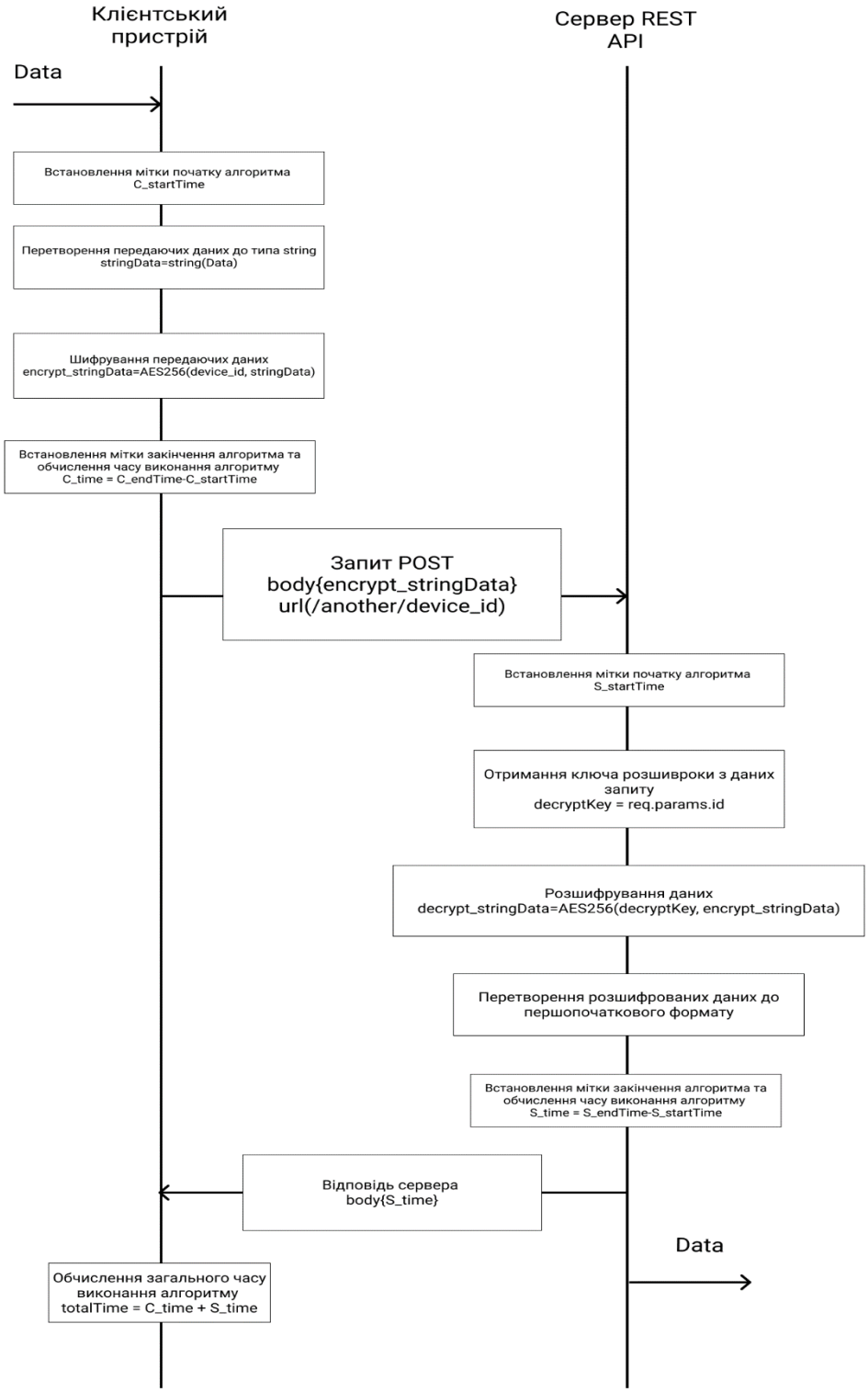


Рис. 3.6 Блок-схема методу передачі на основі AES256 для обчислення часу виконання алгоритму

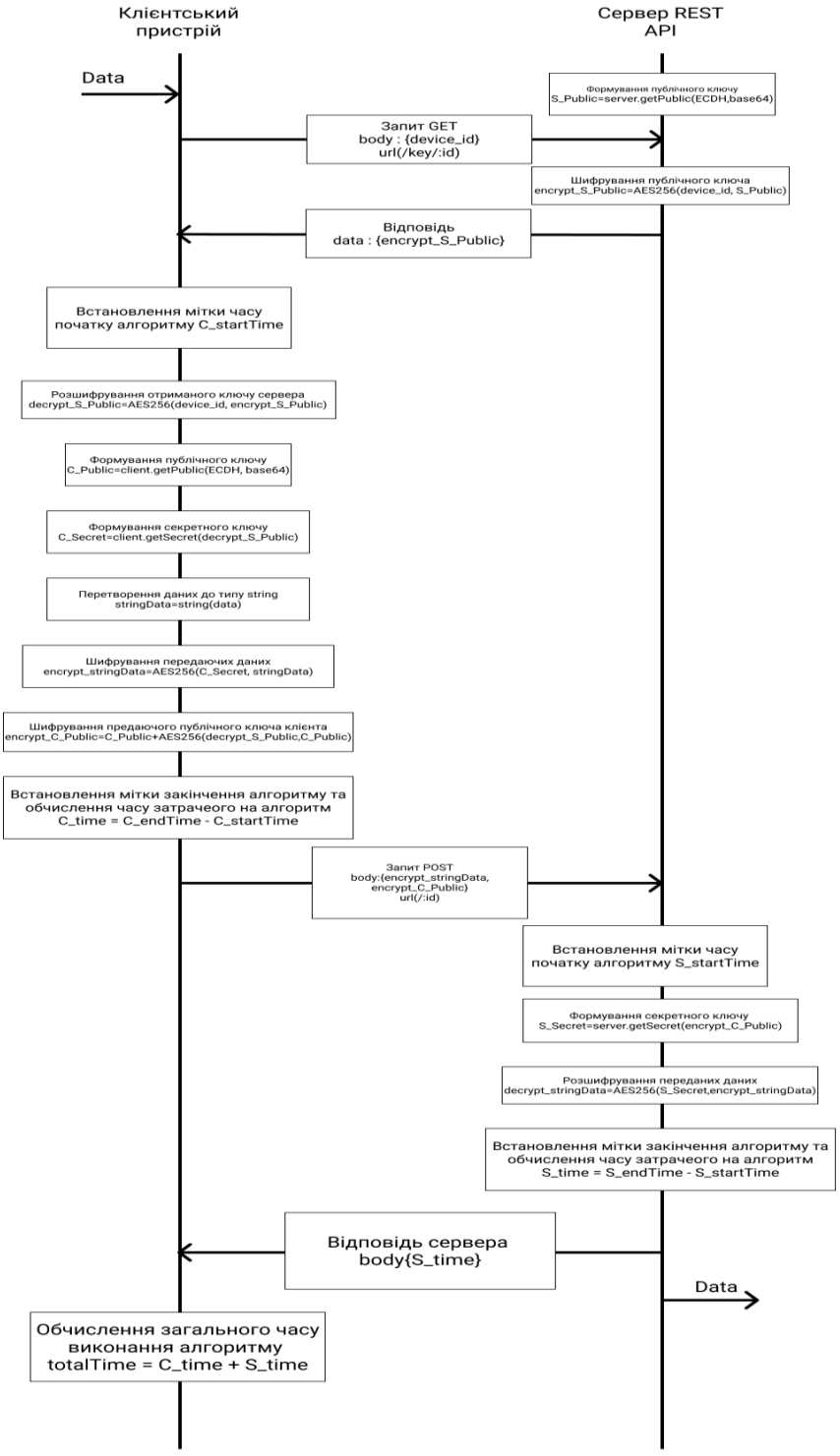


Рис. 3.7 Блок-схема запропонованого методу передачі даних для обчислення часу виконання алгоритму

### 3.2 Аналітична оцінка запропонованого рішення

В даному розділі проаналізовано розглянуті методи натурного моделювання з попереднього розділу.

Аналіз буде проводитись за основним критерієм захисту даних за допомогою шифрування криптографічна стійкість, під криптографічною стійкістю ми маємо наувазі характеристику криптографічного алгоритму відмовляти в криптоаналізі. Криптостійким вважається той алгоритм, який для успішної кібератаки потребує від противника недосяжних обчислювальних ресурсів, обсягу перехоплених відкритих та зашифрованих повідомлень передачі чи ж такого часу декодування, що по його закінченню цього процесу інформація буде вже не актуальна, і т. д. У велікій кількості випадків крипостійкість не можна довести математичними дослідженями, можна тільки довести уразливість криптографічної системи.

Основним показником криптостійкості є рівень криптостійкості. Рівень криптостійкості – величина яка позначає здатність криптостійкості алгоритму, пов'язаний з обчислювальною складністю виконання успішного процесу кібератаки на криптографічну систему. Частіш за все рівень криптостійкості системи вимірюється в бітах. N-бітний рівень криптостійкості криптографічної системи означає, що для її ураженю потрібно виконати n кількість обчислювальних операцій. Розглянемо приклад такий, якщо симетрична криптографічна система зламуються не швидше, ніж за повний перебір значень n-бітного ключа, то визначено, що рівень криптостійкості дорівнює n.

Виходячи з експеременту який ми провели раніше та експертної оцінки ми можемо оцінити рівень криптостійксоті в bit для усіх трьох методів.

Для всіх трьої методів був визначений рівень криптостійкості та складена таблиця порівняння.

Таблиця 3.2

Результати обчислення рівню криптостійкості.

| | Метод без шифрування | Метод з шифруванням AES256 | Метод Еліптичної криптографії | Запропонований метод |
|---|---|---|---|---|
| Рівень криптостійкості, bit | 0 | 256 | 256 | 512 |

**Висновки:**

1. Проведено натурне моделювання запропонованого методу та порівняльних методів. Проведено експеримент порівняння методів за часом виконання алгоритму шифрування, та дешифрування повідомлень.

2. Проведено аналітична оцінка запропонованого методу, та обчислений рівень криптостійкості для запропонованого методу.

# ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

1) Проведено огляд та аналіз проблеми захисту інформації в мережі інтернету речей, за рахунок яких було обрано напрямок вирішення проблеми захисту інформації на рівні програмного забезпечення.
2) Проаналізовано існуючі рішення захисту в мережах Інтернету речей, на основі чого було обрано прототип методу захисту на основі проміжного програмного забезпечення.
3) Удосконалено метод захисту інформації за рахунок впровадження REST API як проміжного програмного забезпечення та еліптичної криптографії, що дозволило підвищити рівень захисту інформації.
4) Виконано натурне моделювання для модифікованого методу на основі REST API та протоколу еліптичної криптографії, що підтверджує працездатність запропонованого рішення. За рахунок аналітичної оцінки, проведеної за допомогою експерименту за часом показано, що показник рівню криптостійкості зріс до 512 bit не не перевищуючи середній час виконання алгоритму 11ms

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1) Securing IoT Devices and SecurelyConnecting the Dots Using REST API and Middleware [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://ieeexplore.ieee.org/document/8777334.
2) Система аутентифікації на базі еліптичних кривих з використанням векторних операцій [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://ela.kpi.ua/bitstream/123456789/25522/1/Albrekht_magistr.pdf.
3) Implementation and integration of efficient ECDH key exchanging mechanism in software based VoIP network [Електронний ресурс] // IEEE. – 2011. – Режим доступу до ресурсу: https://ieeexplore.ieee.org/document/5972416.
4) Анализ проблем безопасности интернета вещей [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://cyberleninka.ru/article/n/analiz-problem-bezopasnosti-interneta-veschey/viewer.
5) Scully P. IoT Security Architecture on the Device and Communication Layers [Електронний ресурс] / Padraig Scully. – 2019. – Режим доступу до ресурсу: https://dzone.com/articles/iot-security-part-1-of-3-architecture-on-the-device-and-communication-layers.
6) Nandi A. An Overview: Security Issue in IoT Network [Електронний ресурс] / A. Nandi, M. Agarwal, D. Samanta // IEEE. – 2018. – Режим доступу до ресурсу: https://ieeexplore.ieee.org/document/8653728.
7) Lightweight Data Security Model for IoT Applications: A Dynamic Key Approach [Електронний ресурс] / M. Kumar, S. Kumar, R. Budhiraja, S. Singh // IEEE. – 2019. – Режим доступу до ресурсу: https://ieeexplore.ieee.org/document/7917128.
8) Урбанович П. П. ЗАЩИТА ИНФОРМАЦИИ МЕТОДАМИ КРИПТОГРАФИИ, СТЕГАНОГРАФИИ И ОБФУСКАЦИИ [Електронний ресурс] / Павел Павлович Урбанович // Белорусский государственный

технологический университет». – 2016. – Режим доступу до ресурсу: https://core.ac.uk/download/pdf/144001465.pdf.

9) Tewari, A., & Gupta, B. B. (2018, January). A robust anonymity preserving authentication protocol for IoT devices. In Consumer Electronics (ICCE), 2018 IEEE International Conference on (pp. 1-5). IEEE
10) Wu, F., Xu, L., Kumari, S., & Li, X. (2017). "A privacy-preserving and provable user authentication scheme for wireless sensor networks based on internet of things security". Journal of Ambient Intelligence and Humanized Computing, 8(1), 101-116.
11) Hsieh, W. B., & Leu, J. S. (2014). A Robust ser Authentication Scheme sing Dynamic Identity in Wireless Sensor Networks. Wireless personal communications, 77(2), 979-989.
12) Chien, H. Y. (2007). SASI: A new ultralightweight RFID authentication protocol providing strong authentication and strong integrity. IEEE transactions on dependable and secure computing, 4(4), 337-340.
13) Razouk, W., Sgandurra, D., & Sakurai, K. (2017, October). "A new security middleware architecture based on fog computing and cloud to support IoT constrained devices".In Proceedings of the 1st International Conference on Internet of Things and Machine Learning (p. 35). ACM
14) Ключевое слово в защите информации // Интернет-портал компании Крипто-про [Електронний ресурс]. – 2000–2016. – Режим доступу до ресурсу:: http://www.cryptopro.ru.
15) Протокол Діффі - Геллмана на еліптичних кривих [Електронний ресурс] // WikiPedia. – 2018. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Протокол_Діффі_-_Геллмана_на_еліптичних_кривих
16) Безвугляк М. С. ЗАХИСТ ІНФОРМАЦІЇ В МЕРЕЖІ ІНТЕРНЕТУ РЕЧЕЙ НА ОСНОВІ REST API / М. С. Безвугляк, В. В. Курдеча //

ПЕРСПЕКТИВИ ТЕЛЕКОМУНІКАЦІЙ / М. С. Безвугляк, В. В. Курдеча. – Київ, 2020. – С. 213–215.

17) Безвугляк М. С. ОСОБЛИВОСТІ ТЕХНОЛОГІЇ REST API В IoT / Максим Сергійович Безвугляк // ПЕРСПЕКТИВИ РОЗВИТКУ ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ ТА СИСТЕМ / Максим Сергійович Безвугляк. – Київ, 2020. – С. 362.